In-the-SPIN

Newsletter of the Boston SPIN

Issue 40 Spring 2001

Editor: Carol Pilch

Editorial

The Spring 2001 edition of *In-the-SPIN* features some insight on process improvement in the '90s from the SPIN Doctor. Take a look at what Judi Brodman, the SPIN Doctor, has to say on this subject, and send her your thoughts and comments. Also, if you were unable to participate in the recent roundtable book discussions, Barbara Purchia and Andrew Hodgdon have provided a report on the roundtable and it's included here in the SPIN Perspectives column. As always, our newsletter provides you with reports from the recent SPIN meetings contributed by Martin Stankard, Sheila Lynch, Judi Brodman, and Carol Pilch.

With this edition, I'm concluding my tenure as Editor of *In-the-SPIN*. I'd especially like to thank all the folks that took the time to contribute to this newsletter during my editorship. It's because of all of the contributors that I've received a lot of positive feedback on *In-the-SPIN*. Please continue to support the newsletter with your contributions as Judi Brodman and Sheila Lynch take the reins as co-editors.

If you're a reader of this newsletter, the Boston SPIN would like your feedback. Consistent with the Boston SPIN charter, *In-the-SPIN* is provided by the Boston SPIN as a means of supporting the free and open exchange of software process improvement experiences and ideas. The Boston SPIN would like to know if the readers' expectations are being met. The steering committee encourages feedback on the newsletter as well as broader participation in the content and production of the newsletter.

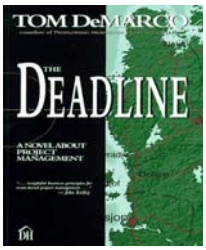
Boston SPIN
Since January 1993

Software Process Improvement Network

SPIN Perspectives

This issue's SPIN Perspectives column features a synopsis of a book discussion that was held at a recent SPIN meeting. The synopsis is contributed by Andrew Hodgdon, Health Physicist and Project Manager at Duke Engineering, and Barbara Purchia, Director of Engineering Operations at Rational Software.

Roundtable – Book Discussion *"The Deadline – A Novel about Project Management," by Tom DeMarco*



You have probably read one of Tom DeMarco's software books. This is his first novel. His character, Mr. Tompkins, is from a series of short stories written in the 1930s by Physicist George Gamow. "In one of these stories Mr. T awoke in a universe where the speed of light was only fifteen miles per hour . . .. he could observe relativistic effects on his bicycle . . . or, in another story, he could see quantum mechanics in action on a billiard table." DeMarco was inspired. "It occurred to me that (fiction) might be used to demonstrate some of the principles of project management."

That kind of demonstration is exactly what "The Deadline" brought to our roundtable book discussion on March 20th. The Deadline is a story about the development of six software products.

Mr. Webster T. Tompkins, an experienced software project manager in a giant telecommunications company, has just been downsized. While he is listening to the company's plans, Lahksa Hoolihan, a beautiful Morovian secret agent, kidnaps him.

Continued on next page

IN THIS ISSUE . . .

Morovia, a former communist country, wants to be the first in the world to export shrink-wrapped software by the year 2000. Tompkin's assignment is to manage a world-class software factory located in Silikon Valejit (sound familiar?). He divides his huge staff of developers into eighteen teams – three for each of the software products. The teams are different sizes and use different methods. They compete against each other and against an impossible deadline.

Mr. Tompkins gets a new boss; the local tyrant threatens his life; and the schedule can't change! What to do? (This character, by the way, is believable as either a communist tyrant or an American boss.) A variety of thinly disguised consultants visit and advise. In the end . . . is the project on schedule? Does the girl get the guy? What did Mr. Tompkins learn? What can we learn? What's next?

At the SPIN meeting on March 20th, over ten people participated in a lively discussion. The chapter on schedule pressure stimulated a good discussion about the net effects of pressure on a project. In the end we all agreed that the journal notes at the end of each chapter were particularly worth reviewing. In fact, we continued the discussion at the May meeting. Participants just read the book and brought an example from their own experience that proved, disproved, or illuminated one of the journal entries.

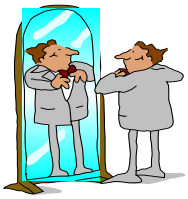
Meeting Summaries

Notes from the February Meeting

Contributed by Martin Stankard, President Productivity Development Group, Inc. Process Consulting and Training, Westford, MA. Martin is a member of the SPIN Newsletter Committee.

Title: Through the Looking Glass: How to do a Post Project Review

Speaker: Barbara Purchia, Director of Engineering Operations at Rational Software



Barbara Purchia provided an excellent and practical guide to getting constructive involvement and valued improvements out of a post project review (PPR). Rather than generate a "whine-o-gram" post mortem report, a well run PPR includes:

- A start to finish summary of project development,
- A description and analysis of project activities, results, lessons learned, and recommendations for improving future development projects
- A focus on the project and process used, never on individuals.

Purchia says that a PPR on a completed project captures an understanding of team dynamics that carry over from project to project. In addition to lessons learned about what went well, what did not go well, and what to improve, PPRs also provide a development team with closure on their project experience. If you have a structured development process you might run a PPR during the project at major milestones, at scheduled dates, or even when needed if things are not going well.

She outlined a highly detailed recipe for cooking up better PPRs.

- Begin by collecting any data, logs, technical or management information, metrics, schedules or test records from the project.
- Plan a 2-hour meeting including representatives from all functions that worked on the project, the project management, and representatives of affected functions. Decide who should be there, the agenda, and meeting room. If you use a questionnaire, tailor it for your project before sending it out. It might cover topics under such headings as "Objectives," "Requirements," and "Project Planning and Risk Management." You may request input from some or all attendees by e-mail.
- Line up the VP in charge or a senior manager in charge of development to sponsor the PPR. Refresh the sponsor on the project milestones and activities. Go over your questionnaire if you plan to use one, and review relevant past PPRs. Ask for the sponsor's input on issue categories to explore (such as requirements and specifications, planning, or communication).
- Ask participants to prepare for the meeting by reviewing their project documentation, project e-mails and by reviewing your PPR questionnaire.
- Have a facilitator run the meeting to stay on time, follow the agenda, engage and protect speakers, and keep the focus on the process and away from blaming and whining. Also, set ground rules and expectations: the PPR will not blame anyone, it is not an in-depth study and it does not cover all the issues.

Purchia also outlined how the facilitator runs the meeting:

- Tape flip chart pages to the meeting room walls and write the potential issue categories at the top of each page. Have a few extra pages to cover issues that arise spontaneously.
- Ask everyone in the meeting to write down his or her issues on Post It Notes, writing one issue per Post It Note using language that lets the issue stand on its own. For example: "Requirements changed at the last minute."
- Collect a Post It issue and all similar or duplicate items and stick them onto the appropriate flip chart page. Continue collecting the issues and placing them on display until everyone is tapped out. Then number the issues.

Next, the facilitator conducts a multi-voting exercise in which participants cast three votes for their top three issues. The winning three issues out of the voting exercise are input to brainstorming for solutions and recommendations. Again, all participants use Post It Notes to come up with solutions and recommendations on the top three issues. These are gathered, discussed, and organized on three flip chart pages spending about 10 to 15 minutes per issue. (Purchia gave an alternative to the above, in which the group breaks down into issue sub-

teams and works in parallel to come up with a solution on each issue.)

After the meeting is over the sponsor collects the metrics, managerial lessons learned and other items that participants may have provided data on by filling in questionnaires. The sponsor writes the PPR document using the following outline:

- Introduction
- Results vs. plan for status, staffing, milestones, cost and quality
- Lessons learned about what went well, what went badly, technical and managerial issues.
- Recommendations for future projects.

Before communicating the PPR document, the sponsor asks participants to review the draft.

The sponsor's greatest responsibility, of course, is to use the PPR as input to planning and executing the next project. Purchia said that her organization's experience with PPRs is that the same issue areas come up. Issues move further down stream as improvements are made. People who participate expect follow-through so sponsors should not let them down, and PPRs have found departmental and project similarities.

PPRs have helped provide information for planning next projects. The comparisons of actual cost have improved project estimation, and they have provided opportunities to share solutions to common problems.

Notes from the March Meeting

Contributed by Carol Pilch, Senior Member of Technical Staff and SEI Authorized Lead Assessor, General Dynamics Communication Systems

Title: The Four R's of Software Process Improvement: Requirements, Reviews, Retrospectives and Results

Speaker: Johanna Rothman, Rothman Consulting Group, Inc.

"Process improvement is not simple; it's not easy; you are changing the culture." This statement was made early on in the March SPIN presentation. It kept my attention, and given Johanna's dynamic presentation style, I'm sure the audience was totally tuned in to hear Johanna's message. Johanna then went on to tell us, "Every process improvement project is different. It has its own context and its own requirements. You need to know and understand your requirements to make decisions about process improvements." Johanna strongly recommends defining your requirements for the process improvement project and using results to derive your requirements.

Context-free questions are a good tool to use to understand your requirements. Ask:

- Who are the clients of the process improvement project?
- What does a highly successful solution look like?
- What is that solution worth to you?
- Why are these results desirable?

Identifying process improvement requirements can be done in the same fashion as deriving product requirements for the project. Johanna recommends creating a matrix of users, attributes, and functions. Translate the results into the question the users will ask: WIIFM (What's in it for me?).

Decide on the attributes that you want to apply to your project (i.e., maintainability, performance, time-to-market, auditable). Consider the attributes and the activities that are critical to the success of your project.

Once you've defined your requirements for process improvement, verify the requirements. Make sure that the business requirements and the "customer" requirements are aligned. It's important to test your requirements with both senior management and the engineers. Make sure that senior management has bought in. Doing this from the beginning helps ensure success. Johanna pointed out that, in this area, lunch is an excellent tool. Have informal conversations with people about the requirements over lunch. Then follow up with formal meetings.

Your next focus is on managing your process improvement requirements. Just as for product requirements, you experience schedule and cost pressure, and requirements change. Understand how you are going to manage changing process improvement requirements. Ask questions when faced with changes:

- What implications does this request have on the customer problem?
- What are the schedule questions?
- Can we still meet the desired business results?

Successful process improvement incorporates reviews. Review requirements and work products for defects. Johanna suggests using Fagan style inspections. Ask what's missing and why because you want to know what's not working. Let people know that you're open to change. Change is desired and expected.

Look at your results. Be able to assess the state of your process improvement efforts. Johanna recommends interim retrospectives and measurements. Interim retrospectives keep you on track. Ask four questions:

- What did we learn (so far)?
- What would we do differently (in the future)?
- What did we do well that we don't want to forget?
- What still puzzles us?

In summary, Johanna stated that process improvement is the act of changing current results to get new results. Change agents need to demonstrate to people that they are open to change and welcome people to find defects in their products.

NOTE: Johanna has published an article on this subject:

Rothman, Johanna. "Four R's of Process Improvement: Requirements, Reviews, Retrospectives, and Results," Crosstalk, May 2000.

Notes from the Joint Boston SPIN/ ASQ April Meeting

Contributed by Sheila Lynch. Sheila is with the Process Improvement Office of MITRE's Information Systems, Infrastructure and Services organization and is now a co-editor of In-the SPIN.

Title: Left Brainers, Right Brainers and No Brainers

Speaker: Rob Peck, CEA (Creative Education Advocate)

Members attending the annual joint Boston SPIN/ASQ meeting on April 19 had ample opportunity to share laughter while Rob Peck, CEA (Creative Education Advocate) entertained all with his presentation, "Left Brainers, Right Brainers and No Brainers." Rob's humorous message about the three 'T's' of Teamwork, Tolerance and Tenacity was dynamically and humorously illustrated by his juggling skills.

Rob's presentation was accompanied by a continuous display of the art of juggling. He showed us that he has mastered juggling, which he defines as achieving a balance between concentration and relaxation, characteristics that are not mutually exclusive. Juggling is about where and how we focus our attention. At some points, you drop the ball. But just admit you messed up and see what you can learn from your mistake. "Never give up," he advised. "Why stop when you are on the brink of mediocrity? Dropping the ball is the same as catching it, only easier. Laugh at yourself and keep your sense of humor."

The metaphor of Rob's message, he emphasized, is that the same is true about achieving high standards of quality. We should concentrate on consistent adherence to the process but relax and stay light. "Laughter induces listening and is the shortest distance between two people." Laughter thus relaxes people and makes them more receptive. He drew a knowing chuckle from all when he defined SPIN as "Seeking Partner in Nitpicking." Just as juggling is a process that takes patience and practice, achieving good software quality is a process that takes teamwork, tolerance and tenacity.

Notes from the May Meeting

Contributed by Judi Brodman. Judi is CEO of LOGOS International, Inc., a local project management and process improvement consulting company. She is a regular contributor to In-the-SPIN as the SPIN doctor and is now co-editor of In-the-SPIN.

Title: Management's Role in Achieving Predictable Software Development

Speaker: Steve Rakitin, President of Software Quality Consulting Inc.

The May SPIN speaker, Steven Rakitin, provided the SPIN members with an overview of a Manager's Role in achieving predictable software development. This presentation covered the following topics: motivation; balancing quality, features, and schedule; and balancing people, process, and product. Steve equated software development with the stock market - as he said - we would be way ahead of the game if we could predict the outcome. The goal of predicable software development is to give your customer what they expect - quality products on time. But software organizations are unable to predict when a product will be released because they lack the discipline necessary to enable them to make that prediction. In order to be competitive in today's global economy, software organizations need to be predictable, credible, and disciplined.

Managers have the ability to change the organization and influence the behavior of the developers, QA, and the project managers. This ability was equated to the ability parents have to influence the behavior of their children - if you want your children to behave in a certain way, you give them an incentive; when they act the way you want them to act, you reward them.

The following can be said of unpredictable organizations:

- They have difficulty planning a new product release
- Staffing projects is difficult for them
- Their customer's are frustrated from promises not kept
- Their employees are frustrated from promises not kept
- They produce many unplanned bug fix releases
- Their product quality is low
- They do not meet their time to market goals consistently
- Their costs are higher than expected
- Their revenue projections are frequently not met
- They over-commit AND under deliver.

Root causes, along with major causes (*), for the unpredictability are:

- Inadequate training (especially software managers)
- Lack of measurement (# of defects, LOC)
- Unrealistic schedules
- Poor project management skills*
- Lack of customer understanding
- Crisis mentality*
- Rewarding of wrong behaviors (heroes)*.

In order to become predictable, software organizations need to:

- Set achievable expectations
- Develop accurate, realistic schedules
- Meet the schedules
- Follow a documented development process (depend on the process)
- Hold people accountable (reward and punishment)
- Proactively manage risk (if not, risk of failure increases)
- Manage internal and external commitments (including sales who make unrealistic schedules with customer).

Predictable organizations have some basics in place: management commitment, risk management and the balance between people, process, and product. These organizations are described as:

- Measuring quality and customer satisfaction regularly (most companies do not do this)
- Measuring employee satisfaction regularly (there is a correlation between customer and employee satisfaction)
- Making effective use of scarce, expensive resources
- Rarely finding themselves in the fire fighting mode
- Having few unplanned bug fix releases
- Following a documented development process
- Actively managing risks
- Under-committing and Over-delivering.

While discussing the economics of software development, the relative cost factor to find and fix defects at each phase of the life cycle was presented ($1 in the Requirements Definition phase to $100 in the Maintenance Phase). An example showing that the cost of finding and fixing a defect pre-release costs half as much as finding and fixing a defect post-release was developed. 'Buggy' code frustrates a manager who wants to allocate resources to new products which generate revenue but instead has to assign resources to an existing product and bug fixing which generates no revenue. Software developers are paid to develop 'buggy' code and then again to fix the 'buggy' code - they are being paid twice to work on the same code.

How do you develop an accurate, realistic schedule? First, you need to understand why your estimates and schedules are wrong most of the time - if you don't know what you are doing wrong, you can't fix it. Managers and developers play ridiculous negotiating games when they are estimating software work, i.e., staff doubling the estimate, management halving the estimate. Software engineers are not trained in methods of estimation but are expected to provide managers with accurate estimates. The three most frequently ignored items impacting estimates and schedules are:

- Interdependencies of tasks
- Non-interchangeability of people
- Known risks.

Most schedules are not based in reality - the world is not a perfect place. Also, once an estimate is made and the schedule developed, no one is held accountable for the estimates or the schedule.

Second, you need to identify and select 'Best Practices' for estimating and scheduling that are appropriate for the organization and require that they be implemented. An example of 'Best Practices' is the use of binary quality gates at the inch-pebble level. On most projects, milestones are used and they are miles apart. By the time the manager is aware that there is a crises, it is too late to do anything about it. Another example of best practices is the use of binary measures - a task is done or not done. There is no partial credit for some part of the task being done. What is meant by 'done' - define the term 'done' so all understand when and if they have completed a task.

And, finally, provide the staff with training in the 'Best Practices' that are chosen by the organization.

What happens when the schedule starts slipping? Usually, the Project Manager panics and starts paring down features; whatever process the project was following is abandoned; activities like reviews (technical and peer) are eliminated; testing is curtailed. The outcome is usually a lose-lose situation for both the software organization and the customer. The project team needs to define what success really means - half the features being implemented?!

What does the manager need to do to create a predicable software development organization? The following actions are given as supporting the Manager's roles in creating a predicable software development organization:

- Work to create a 'process-oriented' culture
- Require a written process that is appropriate and flexible (tailorable)
- Require that the staff who will be following the process be actively involved in creating it
- Ensure that the process is well understood by the staff
- Hold project teams accountable for following the process
- Require that the process be reviewed periodically and changed based on process measures.

But most importantly, the Manager should not 'ram' the process down the throats of the staff; the staff needs to be actively involved in developing the process that they will follow.

Dear SPIN Doctor

Are we doomed…

by Judi Brodman, 'the SPIN doctor'

There is at least one time a year when we feel the need to revisit the past - our birthday. As someone once said, "Those who do not know (revisit) the mistakes of the past are doomed to repeat them" - and once is enough for me! This year I thought I'd look back with you on Software Process Improvement (SPI) and the lessons learned from some of the companies and organizations that have traveled the SPI road over the last 10+ years - companies of varying sizes, producing different software products, and having various management styles and structures. Their SPI goals seemed identical on the surface – to improve the way that they develop their software - but their

motivations were very different. I thought, as the last column of this SPIN year, we could look at these companies, their goals, motivations, approaches and the results of their SPI programs and maybe come away with some 'lessons learned' from the 'mistakes of the past'.

In the early 90's, many companies were undertaking software process improvement because of mandates from their customers, i.e., the US Air Force. If companies wanted to compete for contracts funded by these customers, they had better be evaluated at a Level 3 maturity level using the Software Engineering Institute's (SEI) Capability Maturity Model (SW-CMM®). Their goal, therefore, was to obtain a Level 3 maturity level at all costs; their motivation was to be able to compete for and win their customer's contracts.

What were the results of these early efforts? Many organizations never reached Level 2, never mind Level 3, as attested to by numerous studies; a few organizations reached Level 3 and continued on to Level 4 and 5.

Why did so few succeed back then? First, the CMM and SPI were new concepts. There were very few people who had read the CMM (500+ pages) and even fewer who understood it. Those of us who had and did, took it literally - as it was initially proposed by the SEI. Every organization had to do every practice - as stated. Second, the motivation for improvement was from outside the company (the customer) not from within the company. Therefore, there was no buy-in from the developers and managers. A group viewed as 'outsiders', the SEPG, was imposing the process as well as reams of documentation on the developers and managers. In some cases, the SEPG virtually spent years writing process descriptions, procedures, and templates describing activities and artifacts as they thought they SHOULD be done by the organization – not as they were being done. The process and documentation requirements were then 'rolled' out onto an organization that had no input and, in most cases, they were rejected out of hand. These elements didn't add up to a formula for success!

Lessons learned:
Dictatorships don't work. The organization, staff members from top to bottom, do not respond positively to Software Process Improvement (SPI) being forced upon them – they need and want to participate, i.e., provide input, review, and comment. More importantly, the organization and projects need the process to be THEIR process - the process they use with the holes plugged - which they can start to improve! No value-added comes from adding unnecessary activities and artifacts to the organization's already heavy workload. Literal interpretation of the CMM does not work for most organizations. An organization needs someone (internal or external) who is well versed in the CMM to interpret and/or tailor the practices of the CMM to fit the organization and projects. It needs to concentrate on satisfying the goals as they are stated in the CMM for each of the Key Process Areas (KPAs).

In the mid-90's, many more organizations initiating SPI were commercial organizations who had heard about the SEI's CMM and wanted to know more about it – especially how it could help them. The motivation was their need to become more competitive in their marketplace; their goals were to improve their time to market, reduce defects in their products and keep their cost down.

What happened to these organizations? Many improved - some organizations reached Levels 2, 3 and above - some improved but not by concentrating on levels but on Key Process Areas such as Project Planning and Project Tracking and Oversight. I always wondered at that time why the commercial organizations were able to improve so much more rapidly than the DoD contractors. Now I think it is pretty obvious that what motivated the each group was very different. One group felt that an outside force was imposing SPI on them and so each and every internal change was a battle. The other group was looking to improve the areas that were holding them back and making them less competitive so they were eager to try new things. Many commercial companies were spending much too much time fire fighting and fixing bugs. The developers were ready to try anything so that they could move on to new development rather than spend their time fixing bugs in delivered code. They wanted better project management and they wanted to reduce the number of delivered bugs. The DoD contractors really didn't have any incentive to improve - the government usually ended up paying for whatever happened and the profit from the contract was limited - so where was the incentive?!

Lessons Learned: Motivation to improve comes from within. Developers want to fix the areas that are problems and are causing them the most frustration. Start with those problems first. Project Planning and Tracking are key to a project's success. Defect prevention ranks way up there for developers who want to spend their time building new products versus fixing bugs in old products. Consultants became much better versed in the CMM and some of us even challenged the philosophy of following the CMM as gospel - heretics that we were at the time! We believed in tailoring the CMM for different size projects and different type organizations. The goals were always sacred - even for us heretics!

Today, many organizations, DoD contractors and commercial companies alike, are using the CMM as a guide to aid them in improving the way they manage and develop software. This approach is working well for them. Organizations define their business goals - what they really want to accomplish. They prioritize their problems, fixing the problem areas that are holding them back from meeting those goals. They work together as a team with a known set of common goals. The goal is no longer just a reach a 'Level' but to be a better software development organization.

Lessons Learned: 'Interpret' the CMM and 'tailor' it for each organization and project as necessary. Use it as a guide. Fix the things that are broken and that cause the most frustration in the organization first. Ease into the CMM with the organization - it is still 500+ pages! Setup an SEPG early - don't let them be viewed as outsiders. Use managers and developers from on-going projects. Bring in a consultant who knows the CMM and can apply it to your organization and your problems. Let them relate the problems being fixed to the CMM - don't burden the organization with knowing the details of the CMM. Make the consultant part of your SEPG. Understanding the CMM and applying it to the organization in

a non-strangling, rational manner is still the most difficult part of SPI. Most organizations fail because they are trying to learn the CMM and fix the organization at the same time. Document your process as you go, fill in the holes, roll everything up, create the policies for the organization as the last act.

In other words, be flexible, listen to your organization, understand your culture, keep your approach simple, and don't force a square peg into a round hole! I'll leave you to think about these lessons learned over the summer - maybe if we do look back, we will not be 'doomed' to repeat our mistakes.

Send your comments and questions to me through the summer. I will hopefully be back in print in September. And now, my usual pre-summer closing…

"Here in the North, summer means a less "formal" way of life, fewer rigid "plans and procedures" - a less "institutionalized" way of life - more cookouts, long walks on the beach at Cape Cod, summer nights of gardening, midnight swims, picnics by a waterfall in Vermont.... I guess life reverts to 'Level 1' for us for a few months.... Enjoyably "chaotic" !!!" Have a wonderful summer.

This column is for you; let's make a difference! Send your comments or questions to "Dear SPIN doctor" at brodman@LOGOS-Intl.com. Sign them or use a "pen-name"- I respect your confidentiality.

"the SPIN doctor"

"The process is daily tribulation for eventual salvation…" - anonymous.

Looking for Interesting Speakers

We are always looking for interesting speakers. If you'd like to speak at Boston SPIN, please review these criteria before sending us an abstract.

Speaker Criteria:

1. The topic must be timely, an issue of concern to our membership.
2. We want to hear about real-world topics. If you have an academic topic, we're interested in how it applies to the real world.
3. If you are interested in creating a panel, please write an abstract, and suggest at least one panelist. We can work with you to find other panelists.
4. The topic should either explain how to *do* something, or bend our brains in another direction.
5. The presenter should be intimately involved with the "hows" of the thing that got done.
6. We are not interested in sales pitches.

If you have information you'd like us to hear, please send an abstract to Anna Allison, anna_allison@yahoo.com.
We developed a speaker checklist so that none of us would have to rely on our short-term memories. Please use the checklist to prepare for your SPIN talk.

Speaker Checklist:

1. 60 days in advance of meeting deliver: 2-paragraph abstract, one paragraph bio, and picture to anna_allison@yahoo.com
2. Within one week of meeting date: If desired, email copy of paper or overheads to heimann@world.std.com so that it is downloadable from the SPIN web page.
3. At the meeting: Speaker provides one copy of overheads to Linda McInnis for our library.
4. Optional, but highly desired: Send a copy of overheads, paper, etc. for our web page as of the day of the meeting. If possible, provide 50-60 copies of overheads at the SPIN meeting. (The attendees really appreciate this.)
5. Optional: If you've written a book and are willing to donate it to SPIN, we'd be happy to offer the book as a door prize by conducting a free drawing.

Boston SPIN

The Boston SPIN is a forum for the free and open exchange of software process improvement experience and ideas. Meetings are usually held on third Tuesdays September - June. Boston SPIN welcomes volunteers and sponsors. There is no charge to attend the meetings
For more information about our programs and events contact:
Linda McInnis
80 Harris Street, Suite 2
Acton, MA 01720
Telephone: (978) 635-9281
Email: Boston_SPIN@yahoo.com

For information about SPINs in general including ***HOW TO START A SPIN*** contact:

Dawna Baird of SEI (412) 268-5539, dbaird@sei.cmu.edu, http://www.sei.cmu.edu/collaborating/spins/spins.start.html

IN THE SPIN is available on our Web page:

http://www.cs.uml.edu/Boston-SPIN.

TO RECEIVE NOTIFICATION OF NEW IN-THE-SPIN ISSUES and Boston SPIN specific notices send email addressed to withall@mediaone.net.

We have 2 separate email lists: one for this newsletter and one containing announcements that we receive from other process organizations and forward out.

IF YOU WANT TO JOIN THE ANNOUNCEMENTS LIST: http://groups.yahoo.com/group/Boston-SPIN-Announcements/join.

Send letters-to-the-editor, and general correspondence to Judi Brodman, brodman@LOGOS-Intl.com.

Send job postings to heimann@world.std.com.

Back issues and other information about Boston SPIN can be found at our WEB HOME PAGE:
http://www.cs.uml.edu/Boston-SPIN/

.